STATE of ARIZONA

Government Information Technology Agency	Statewide STANDARD	TITLE: Data Modeling
	P740-S740	Effective Date: August 15, 2003

1. AUTHORITY

The Government Information Technology Agency (GITA) shall develop, implement and maintain a coordinated statewide plan for information technology (IT) (A.R.S. § 41-3504(A (1))), including, the adoption of statewide technical, coordination, and security standards (A.R.S. § 41-3504(A (1(a)))).

2. PURPOSE

Data modeling is the formalization and documentation of existing processes and events that occur during application software design and development. It simplifies the complex process of software design, making a "blueprint" for construction. The purpose of this standard is to establish guidelines for budget unit and State implementations of data modeling that facilitate common, interoperable representations and descriptions of data and information that is collected and managed.

3. SCOPE

This applies to all budget units. Budget unit is defined as a department, commission, board, institution or other agency of the state organization receiving, expending or disbursing state funds or incurring obligations of the state including the board of regents and the state board of directors for community colleges but excluding the universities under the jurisdiction of the board of regents and the community colleges under their respective jurisdictions and the legislative or judicial branches. A.R.S. § 41-3501(2).

The Budget Unit Chief Executive Officer (CEO), working in conjunction with the Budget Unit Chief Information Officer (CIO), shall be responsible for ensuring the effective implementation of Statewide Information Technology Policies, Standards, and Procedures (PSPs) within each budget unit.

4. STANDARD

Data modeling is the graphical mapping of data/information in its relations to IT software application systems, sub-systems, and processes that gather, manage, transform, and communicate such data/information. Data modeling is an important component of systems development and re-engineering that can ensure that business functionality is complete and correct, end-user needs are met, and program design supports requirements for scalability, robustness, security, extensibility, and other characteristics, before implementation in programming code renders changes difficult and expensive to make.

4.1. In accordance with the implementation criteria defined in *Statewide Policy P740, Data/Information Architecture*, when required, budget units shall complete, and subsequently update throughout the development cycle of the IT project, a high-level physical data flow diagram (DFD) for the IT project's

software application, such as illustrated in *Attachment A, Data Modeling Symbols, Definitions, and Samples*.

4.1.1. Physical DFDs are required for top-level systems (parents) and subsystems (children) and shall depict the IT project's data flows, data stores, data processes, and external entities (data sources/destinations).

- A physical DFD provides a holistic, graphical representation of the flow of data into and out of processes implemented by software application systems and their respective subsystems.
- A physical DFD focuses on the data being passed through processes to identify origination, transformation, and destination of data; data stores; and entity relationships as a result of business events and activities.
- The physical DFD depicts the way a software application system will implement and execute the processes, independent of specific hardware and software.

4.1.2. Logical DFDs shall be required only when their creation is determined to benefit the IT project, budget unit, community of interest, or the State. The State CIO, in agreement with the Budget Unit CIO, shall make the determination of benefit as well as the mutually agreed-upon deliverables.

- Logical DFDs are a further breakdown (decomposition analysis or leveling of subsystems) to more precisely understand the transformation and/or manipulation of data that is otherwise not identified in the physical DFD parent process.

4.1.3. Physical DFDs (and logical DFDs, when required) for new, custom-developed application software and/or re-engineering projects shall be completed for the "To-Be" software application systems and subsystems.

4.1.4 "As-Is" data models for custom-developed application software and/or re-engineering projects shall be completed at the discretion of the budget unit CIO.

4.1.5 Physical DFDs depicting the "As-Is" software application systems and subsystems shall be required for new, commercial-off-the-shelf (COTS) or government-off-the shelf (GOTS) application software whenever significant modifications are made. Additions or customizations of reports, views, queries, etc. are not considered significant modifications.

4.1.6 Data modeling is not required whenever the budget unit implements, without significant modification, a new, commercial-off-the-shelf (COTS) or government-off-the shelf (GOTS) application software system.

4.1.7. The statewide method for data flow diagramming within the constructs of data modeling is based on the Yourdan/DeMarco methodology. Icons and relationships used in the physical DFD shall conform to the Yourdan/DeMarco methodology (see *Attachment A, Data Modeling Symbols, Definitions, and Samples*).

4.2. CONSISTENT DATA MODELING TOOLS SHALL BE USED WITHIN EACH BUDGET UNIT.

4.2.1. The success of a system design or re-engineering process is dependent upon the skills of the analyst/designer. Data modeling techniques and tools capture and translate complex system designs into easily understood representations of the data flows and processes of an organization.

4.2.2. Consistent tools facilitate changes and enhancements to models and resulting application systems, as well as advancing the sharing of information and promoting reusable program code.

4.2.3. Data modeling tools used in conjunction with relational and hierarchical database technologies shall be Extensible Markup Language (XML) compliant.

- XML is the universal format for structured documents and data on the Internet. XML is a set of rules (guidelines or conventions) for designing text formats to structure data. XML is extensible, platform-independent, and it supports internationalization and localization. XML is intended for the storage and manipulation of text making up human-readable documents like Web pages.
- Extensible Markup Language Metadata Interchange Format (XMI) is a model-driven XML integration framework for defining, interchanging, manipulating, and integrating XML data and objects.
  - XMI-based standards are in use for integrating tools, repositories, applications, and data warehouses.
  - XMI specifies an open information interchange model that is intended to give developers working with object technology the ability to exchange programming data over the Internet in a standardized way, thus bringing consistency and compatibility to applications created in collaborative environments.
  - XMI allows translation of a UML model from one tool into a repository or into another tool for refinement as the next step in the development process.

4.2.4. Data modeling tools used in conjunction with object-oriented database technologies shall be Unified Modeling Language (UML™) and Extensible Markup Language (XML) compliant.

- UML is the industry-standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems. UML standardizes the description of software designs, particularly object-oriented designs.
- UML is methodology-independent. Regardless of the modeling methodology used to perform analysis and design, UML can be used to express the results.
- UML is supported by a broad base of industry-leading companies. UML merges the notations used by the three most popular analysis and design methodologies, Booch, Object-Oriented Software Engineering (OOSE) (use-cases), and Object Modeling Technique

(OMT), to produce a single, universal modeling language that can be used with any method.

- UML is a rigorous modeling language that standardizes the description of software designs, particularly object-oriented designs.

5. DEFINITIONS AND ABBREVIATIONS

5.1. "As-Is" refers to the current business and technology processes being addressed.

5.2. "Commercial off-the-shelf" (COTS) application software is a product that is used "as-is." COTS software is designed to be easily installed and to interoperate with existing system components.

5.3. "Custom-developed" application software is defined as software that is specifically designed and programmed for an individual budget unit or customer.

5.4. **"**Government off-the-shelf" (GOTS) application software is defined as software developed for a government agency with funding and specification from the agency that is made available to other government agencies. GOTS includes technology/system transfers from other government agencies.

5.5. "To-Be" refers to the new business and technology processes to be implemented.

5.6. Refer to the Glossary of Terms located on the GITA website at http://www.gita.state.az.us/policies_standards for additional definitions and abbreviations.

6. REFERENCES

6.1. A. R. S. § 41-621 et seq., "Purchase of Insurance; coverage; limitations, exclusions; definitions."

6.2. A. R. S. § 41-1335 ((A (6 & 7))), "State Agency Information."

6.3. A. R. S. § 41-1339 (A), "Depository of State Archives."

6.4. A. R. S. § 41-1461, "Definitions."

6.5. A. R. S. § 41-1463, "Discrimination; unlawful practices; definition".

6.6. A. R. S. § 41-1492 et seq., "Prohibition of Discrimination by Public Entities."

6.7. A. R. S. § 41-2501 et seq., "Arizona Procurement Codes, Applicability."

6.8. A. R. S. § 41-3501, "Definitions."

6.9. A. R. S. § 41-3504, "Powers and Duties of the Agency."

6.10. A. R. S. § 41-3521, "Information Technology Authorization Committee; members; terms; duties; compensation; definition."

6.11. A. R. S. § 44-7041, "Governmental Electronic Records."

6.12. Arizona Administrative Code, Title 2, Chapter 7, "Department of Administration Finance Division, Purchasing Office."

6.13. Arizona Administrative Code, Title 2, Chapter 10, "Department of Administration Risk Management Section."

6.14. Arizona Administrative Code, Title 2, Chapter 18, "Government Information Technology Agency."

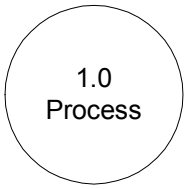
6.15. State of Arizona Target Data/Information Architecture.

6.16. Statewide Policy P100, Information Technology.
6.17. Statewide Policy P700, Enterprise Architecture.
6.18. Statewide Policy P740, Data/Information Architecture.

7. ATTACHMENTS

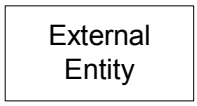
A. Data Modeling Symbols, Definitions, and Samples.

ATTACHMENT A. Data Modeling Symbols, Definitions, and Samples.



Process Symbol: represents an activity that transforms or manipulates the data (combines, merge, calculates, creates, converts, prints, etc).

Data Store: represents a repository of data not moving. Physical files such as databases, desk files, mag/cartridge tapes, diskettes, CDs, etc.

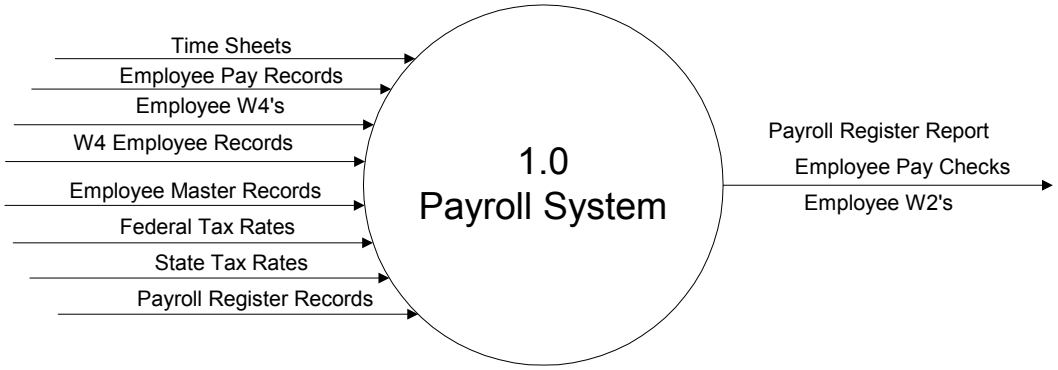


External Entities: are external to the system being modeled and beyond the area of influence and control.

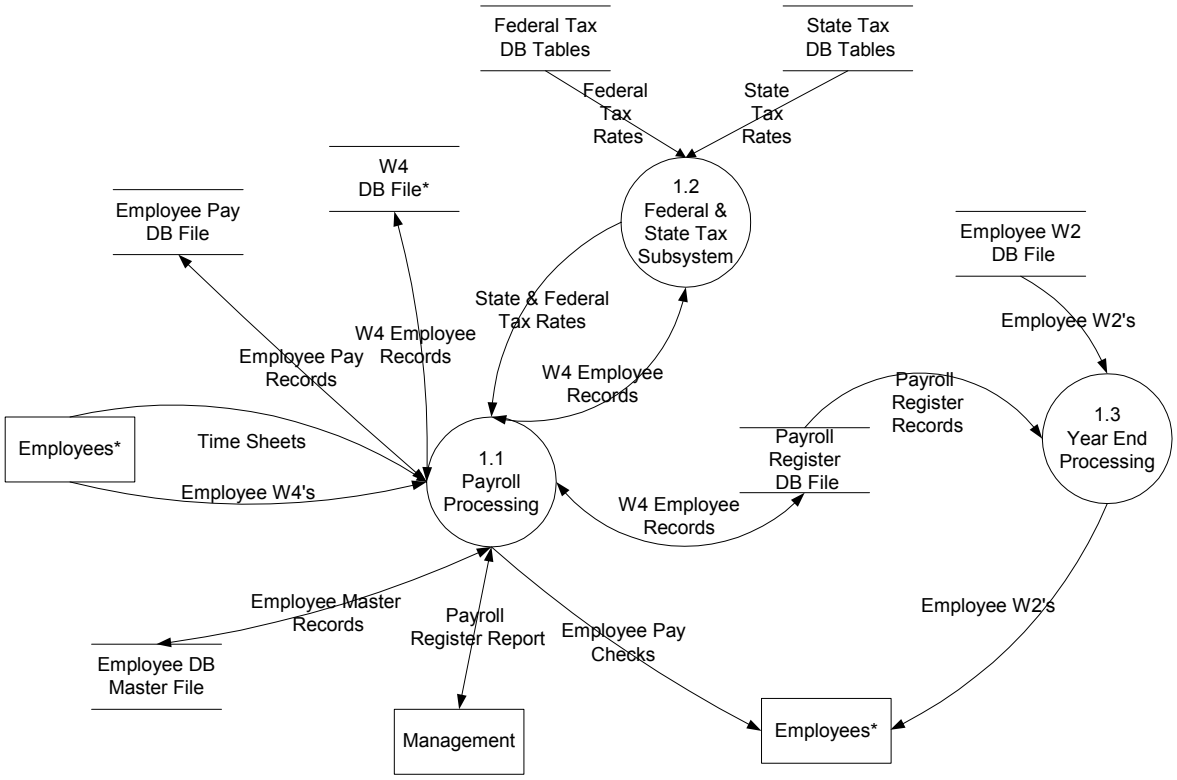
Data Flows: represents the movement of data not control (forms, files, transactions, reports, documents, etc). Data flowing both directions between two symbols would be noted by placing arrow heads on both ends of the connector.

Sample Data Flow Diagrams Showing Relationships

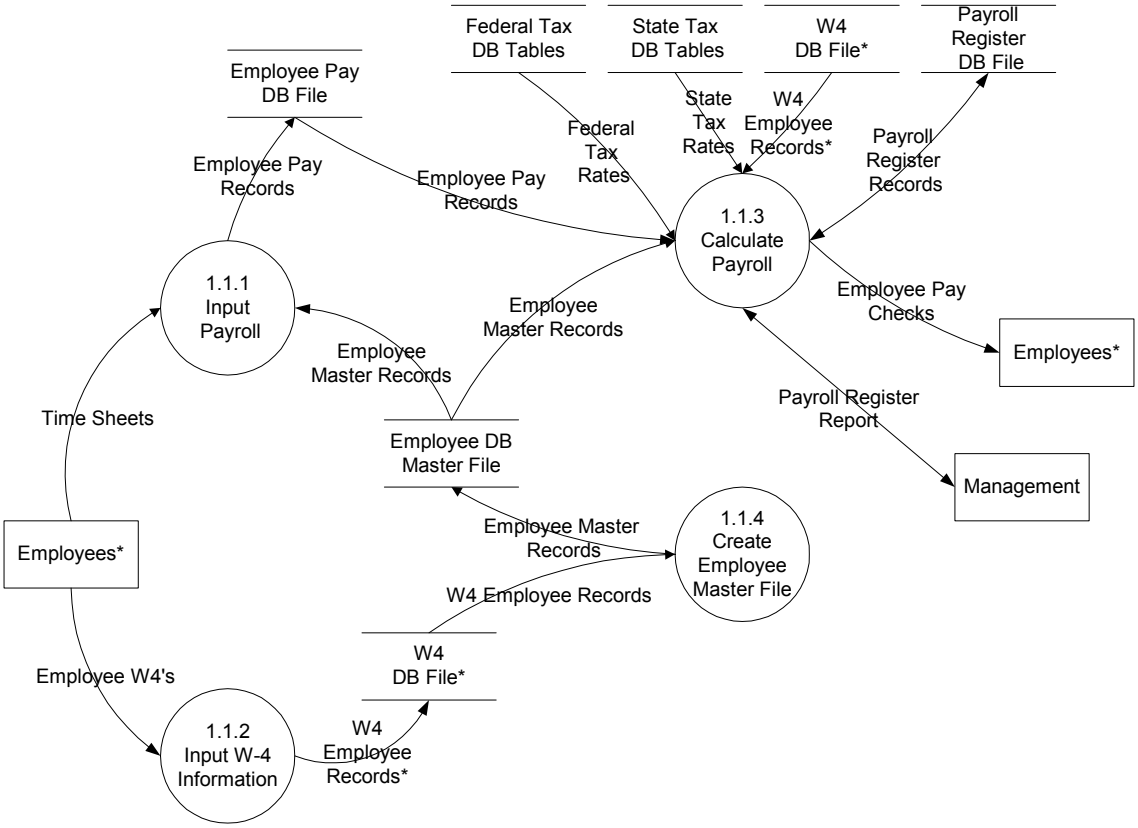
System Context Diagram



Physical DFD -1st Level Decomposition of 1.0 Payroll System



Logical DFD - 2nd Level Decomposition of 1.1 Payroll Processing



Note: All data flows balance, i.e., they are identified at all three levels of modeling: 1) the System Context Diagram, 2) the Physical DFD - 1st Level Decomposition, and 3) the Logical DFD - 2nd Level Decomposition.